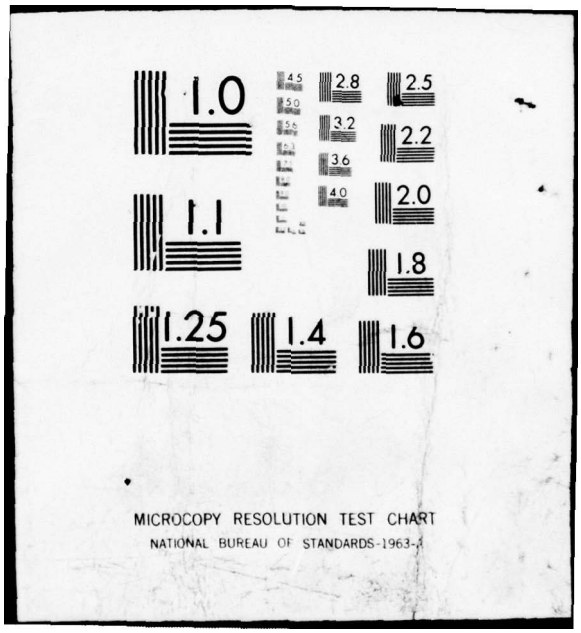
AD-A041 887 DEFENSE SYSTEMS MANAGEMENT COLL FORT BELVOIR VA F/G 5/1
SOFTWARE MANAGEMENT: APPLYING HARDWARE DISCIPLINES TO SOFTWARE --ETC(U)
NOV 76 L A ANDERSON

UNCLASSIFIED NL

| OF |
AD
A041887

END

DATE
FILMED
8-77


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(1)

# DEFENSE SYSTEMS MANAGEMENT COLLEGE



## PROGRAM MANAGEMENT COURSE
## INDIVIDUAL STUDY PROGRAM

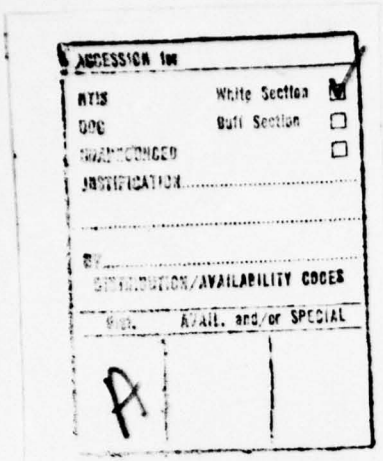
SOFTWARE MANAGEMENT:
APPLYING HARDWARE DISCIPLINES
TO SOFTWARE MANAGEMENT

STUDY PROJECT REPORT
PMC 76-2

Lee Arthur Anderson
Lt Col USAF



FORT BELVOIR, VIRGINIA 22060




# SOFTWARE MANAGEMENT:

# APPLYING HARDWARE DISCIPLINES

# TO SOFTWARE MANAGEMENT

Study Project Report

Individual Study Program

Defense Systems Management College

Program Management Course

Class 76-2

by
Lee A Anderson
It Col USAF

November 1976

Study Project Advisor
LCDR Susan Anderson

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

## REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

| | |
|---|---|
| 1. REPORT NUMBER | 2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER |
| 4. TITLE *(and Subtitle)*<br>SOFTWARE MANAGEMENT: APPLYING HARDWARE DISCIPLINES TO SOFTWARE MANAGEMENT | 5. TYPE OF REPORT & PERIOD COVERED<br>Student Project Report 76-2<br>6. PERFORMING ORG. REPORT NUMBER |
| 7. AUTHOR(s)<br>LEE A. ANDERSON | 8. CONTRACT OR GRANT NUMBER(s) |
| 9. PERFORMING ORGANIZATION NAME AND ADDRESS<br>DEFENSE SYSTEMS MANAGEMENT COLLEGE<br>FT. BELVOIR, VA 22060 | 10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS |
| 11. CONTROLLING OFFICE NAME AND ADDRESS<br>DEFENSE SYSTEMS MANAGEMENT COLLEGE<br>FT. BELVOIR, VA 22060 | 12. REPORT DATE<br>76-2<br>13. NUMBER OF PAGES<br>34 |
| 14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)* | 15. SECURITY CLASS. *(of this report)*<br>UNCLASSIFIED<br>15a. DECLASSIFICATION/DOWNGRADING SCHEDULE |

16. DISTRIBUTION STATEMENT *(of this Report)*

UNLIMITED

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

SEE ATTACHED SHEET

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

SEE ATTACHED SHEET

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

DEFENSE SYSTEMS MANAGEMENT COLLEGE

STUDY TITLE: SOFTWARE MANAGEMENT:
APPLYING HARDWARE DISCIPLINES TO SOFTWARE MANAGEMENT

STUDY PROJECT GOALS: To understand, from a systems acquisition viewpoint, how certain proven hardware management disciplines can be used to improve software management. To evaluate and explain the use of configuration management, data management, and design reviews in the life cycle of software management.

STUDY REPORT ABSTRACT: After an introduction, this study report discusses the acquisition life cycle. The report outlines how a life cycle should be used to manage the acquisition of software. Configuration management is introduced by relating the configuration management baselines to the life cycles. It further discusses the configuration management functions of configuration identification, configuration control, and configuration status accounting! The report strongly supports the use of configuration management as a discipline in the management of software. The report next discusses the use of technical reviews and audits in the management of software. Finally the report discusses what data must be procured to support a software system. The general conclusion of the report is that hardware disciplines can and should be adapted and used in the management of software.

The author used documents published by the Department of Defense and the United States Air Force as guidance in presenting the material. However,the author attempted to write the report in general terms so that Navy and Army personnel, along with Air Force personnel could read and understand the report. Also, the author hopes that persons familiar with the acquisition of hardware, but not software, could read this report and begin to understand software management.

SUBJECT DESCRIPTORS: Software Management
Configuration Control
Life Cycle
Baseline Management

| NAME, RANK, SERVICE | CLASS | DATE |
|---|---|---|
| Lee A. Anderson, Lt Col, USAF | PMC 76-2 | November 1976 |

## Executive Summary

The purpose of this study project was to further the author's understanding of software management and to discuss, from a systems acquisition viewpoint, how certain proven hardware management disciplines can be adapted and used to improve software management.

The author used documents published by the Department of Defense and the United States Air Force as guidance in presenting the material. However, the author attempted to write the report in general terms so that Navy and Army personnel, along with Air Force personnel could read and understand the report. Also, it is hoped that persons familiar with the acquisition of hardware, but not software, could read this report and begin to understand software management.

The Department of Defense and the Military Services are very concerned with the acquisition of software. They have published many directives regulations and standards that guide the acquisition of software. These documents describe how acquisition life cycles, baselines, configuration management, etc. can be used to manage software.

Program managers can use the same disciplines to manage software as they use to manage hardware. Using these management disciplines should result in the satisfactory acquisition of software.

ACKNOWLEDGEMENTS

To LCDR Susan Anderson, my project advisor, I express grateful appreciation for her encouragement, support and excellent advice.

To Lt Col John Marciniak and Capt Maurice Comeau, I express appreciation for the help they provided in gathering background material for the project.

To Mrs Nancy LeBoeuf, I express appreciation for her excellent typing support.

TABLE OF CONTENTS

SECTION I

# INTRODUCTION

## Purpose of the Study Project

During recent years the Department of Defense (DOD) and the service components have recognized the need for improved management of software during acquisition. This need was outlined by Deputy Assistant Secretary of Defense Jacques S. Gansler in the October 1975 issue of the Defense Management Journal which was devoted to software. His introductory comments included the following remarks:

> In recent months, managers in Defense and industry have been challenged by three important weapon system software issues: the lack of sufficient control over rapidly growing software expenditures, the lack of sufficient research and development in software production, and the need for major improvements in weapon systems software management (10:1). *

In an attempt to improve the management of software, the Department of Defense and the service components have initiated three types of action during the past five years: they have held conferences on software; and they have published new directives and regulations concerning software. The reports from some of the studies are listed in the bibliography to this report. One of the new directives is DOD Directive 5000.29, 26 April 1976, Subject: Management of Computer Resources in Major Defense Systems. The general policy stated in this directive is as follows:

> Annual expenditures by DOD on the design, development, acquisition, management, and operational support of computer resources embedded within, and integral to weapons, communications, command and control, and intelligence sensor systems

* This notation will be used throughout the report for sources of quotations and major references. The first number is the source listed in the bibliography. The second number is the page in the reference.

> are measured in the billions of dollars. Unreliability, particularly of software, diminished DOD mission effectiveness in many major Defense systems.
>
> Computer resources in Defense systems must be managed as elements of subsystems of major importance during conceptual, validation, full-scale development, production, deployment, and support phases of the life cycle, with particular emphasis on computer software and its integration with the surrounding hardware (12:2).

Computer resources consist of two main elements: hardware and software. The cost of hardware (computer equipment) has generally decreased over the years as the industry has learned new techniques and progressed through several generations of computer equipment improvements. The cost of software (computer programs) has increased instead of decreased. In certain cases, even after spending a considerable amount of funds, the development of computer programs has been unsatisfactory and the final programs have not accomplished their intended purpose. Therefore, I intend to discuss software management in this report. Specifically, can proven hardware disciplines be adapted and applied to the software acquisition process?

## Goals of the Study Report

The author of this report had five years of experience managing software for the Air Force Satellite Control Facility in support of military satellites. My experience has been primarily in the deployment phase. My division had the responsibility to accept all computer programs at the end of the development/integration phases, to verify that the computer programs were ready for operational use, and to analyze all problems that occured during the operational use of the computer programs.

I had two goals when choosing this study subject. The first goal

was to determine what current overall guidance for managing the acquistion of software had been published by DOD and the United States Air Force. The second goal was to further determine if hardware techniques could be used in software management.

My overall objective in organizing and writing the report was to provide a report that a person not experienced in software management could read and understand. I believe that software management is no more difficult that hardware management. A manager new to software management should be successful if he applies the proper disciplines.

## Definitions

Computer Data. Basic elements of information used by computer equipment in responding to a computer program (12:Encl 1).

Computer Equipment. Devices capable of accepting and storing computer data, executing a systematic sequence of operations on computer data or producing control outputs. Such devices can perform substantial interpretation, computations, communication, control and other logical functions (12:Encl 1).

Computer Hardware. Used interchangably with computer equipment in this report.

Computer Program. A series of instructions or statements in a form acceptable to computer equipment, designed to cause the execution of an operation or series of operations. Computer programs include such items as operating systems, assemblers, compilers, interpreters, data management system, utility programs such as payroll, inventory control, operational flight, strategic, tactical, automatic test, crew simulator, and engineering analysis programs. Computer programs may be either machine dependent or machine independent, and may be general purpose in nature or be designed to satisfy the requirements of a specialized process of a particular use (12:Encl 1).

Computer Program Maintenance. The maintenance of a computer program is defined to be any modification to the instruction of an operating digital computer program or to the documentation that describes the computer

program. This maintenance may be accomplished for either corrective or improvement purposes and is normally connected with the deployment phase of the life cycle.

Computer Resources. The totality of computer equipment, computer programs, computer data, associated documentation, personnal, and supplies(12:Encl 1).

Computer Software. A combination of associated computer programs and computer data required to enable the computer equipment to perform computational or control functions (12: Enc 1).

Configuration. The functional and/or physical characteristics of hardware/software as set forth in technical documentation and achieved in a product (14:41).

Configuration Control. The systematic evaluation, coordination, approval or disapproval, and implementation of all approved changes in the configuration of a CI after formal establishment of its configuration identification (14:41).

Configuration Item (CI). An aggregation of hardware/software, or any of its discrete portions,which satisfies an end use function and is designated by the Government for configuration management. CIs may vary widely in complexity, size and type (14:41).

Configuration Management. A discipline applying technical and administrative direction and surveillance to (1) identify and document the functional and physical characteristics of a configuration item, (2) control changes to those characteristics, and (3) record and report change processing and implementation statics (14:42).

Contract Data Requirements List (CDRL). A contract form, DD 1423, listing all technical data items and status data items, selected from an Authorized Data List, to be delivered under the contract.

Data (Technical). The means for communication of concepts, plans,descriptions, requirements, and instructions relating to technical projects, material, systems, and services. These may include specifications, standards, engineering drawings, associated lists, manuals, and reports, including scientific and technical reports; they may be in the form of documents, displays, records, punched cards, and digital or analogy data (14:42).

Data (Status). The means for communication of status on a program/project. These may include reports on funds, cost expenditure,cost forecasts, schedules, schedule forecasts, agendas for meetings, etc.

Documentation. The specifications, reports, plans, and procedures used to document and support computer programs(6:4).

Embedded. Adjective modifier; integral to, from the design, procurement, and operations point of view exposed in DOD Directive 5000.1 (12:Encl 1).

Hardware System. The totality of a system of hardware including the hardware, associated documentation, personnel and logistics.

Software Engineering. Science of design, development, implementation, test, evaluation, and maintenance of computer software over its life cycle (12: Encl 1).

Software System. The totality of a system of software including the software, associated documentation, personnel and logistics.

## Organization of the Study Report

In addition to this introductory section, the report has been organized into four sections and a summary section. The reader will possibly want to read the summary section before he reads the other four sections.

Section II deals with the life cycle. First, the acquisition life cycle is discussed; secondly, additional life cycles as they relate to software are discussed; this is followed by the author's recommended life cycle. Finally the section brings out some important items to consider during each phase of the life cycle.

Section III deals with configuration management. This section begins with some general comments, followed by some comments on baseline management. Finally the configuration management functions of configuration identification, configuration control, and configuration status accounting are discussed.

Section IV deals with technical reviews and audits. This section begins with general comments, followed by a discussion of each type of

formal technical review and formal audit.

Section V deals with the management of technical and status data. This section begins with general comments, followed by a section that discusses the need for data/documentation. Finally the section discusses some specific software data items.

Section VI is the summary containing conclusions and recommendations.

SECTION II

# THE LIFE CYCLE

## The Acquisition Life Cycle

The acquisition process normally uses a life cycle to provide the basis for categorizing program management activities. In general the cycle is divided into five phases: conceptual, validation, full-scale development, production, and deployment phases. This is the life cycle commonly used to acquire hardware. A brief description of each phase is provided in the paragraphs below.

During the conceptual phase, the technical, military and economic bases are established through comprehensive studies, experimental development and concept evaluation. The management approach is also formulated during this phase. During the validation phase the major program characteristics are validated and refined through studies, system engineering and preliminary development, test and evaluation. Program risks and costs are assessed, resolved or minimized.

During the full-scale development phase, the system is designed, fabricated, tested and evaluated. The test results demonstrate that the system meets the stated performance requirements. Costs are continually assessed. During the production phase, the system is produced and delivered as an effective, economical, and supportable system. Transition from the acquisition to the operation and support commands begins. During the deployment phase, the system is used operationally. This phase terminates when the system is removed from the operational inventory.

## Other Software Life Cycles

In researching documentation, the author noted that several documents listed the names for the steps or phases for the acquisition of software to be different from the names used in normal hardware acquisition cycle. Three of these software life cycles (there are more) are discussed below.

AF Regulation 800-14 lists the life cycle for computer program development in six phases: analysis, design, coding and checkout, test and integration, installation, and operating and support phases (1:2-3). These phases can be related to the major acquisition phases. For example, the analysis phase corresponds partly to the conceptual phase, and the operational and support phase corresponds to the deployment phase.

In the past, the Space and Missile Systems Organization used a milestone system for development of computer programs. These development milestones are outlined in SSD Exhibit 61-47 B (5:3). The milestones were named or expressed in terms of the product available at the end of each milestone. The eight milestones were: design criteria, implementation concept and test plan, interface specifications, design and acceptance specifications, computer programs, computer program subsystem validation plan, computer program operation instructions, computer program subsystem and model capabilities. This is the system that the Air Force Satellite Control Facility was using when the author first became involved with software. These milestones can be related to the other life cycles. For example, the design criteria is the product of a conceptual

phase.

The Air Force has recently written a document for use as a guide in the development of software. This guide, with the title "Air Force ADP Software Project Manual", lists the following five development phases: conceptual, definition, development, integration, and operational (3:2-1). These phases can be directly related to the five major acquisition phases. For example, the definition phase corresponds to the validation phase.

Authors Recommended Software Life Cycle

The acquisition life cycle can be used to define the phases of software acquisition as well as hardware acquisition. However, the fourth phase, production does not have a significant meaning in software acquisition. To make additional copies of the master computer program is very simple. In software acquisition this phase consists mainly of testing, not production. In hardware acquisition both production and testing of the product is a significant effort. Therefore, for software acquisition it is suggested that the life cycle consists of the following five phases: conceptual, validation, development, integration, and deployment. The author believes that by using these names a manager new to software acquisition can easily relate software acquisition to his experience in hardware acquisition.

Figure I displays how the different life cycles can be compared to each other. A manager should evaluate his situation, choose an appropriate life cycle, and tailor his situation to that cycle. It isn't important what name is given to each phase; however, it is extremely important that a life cycle be used to manage the software project. A life cycle helps the manager think through the software acquisition process.

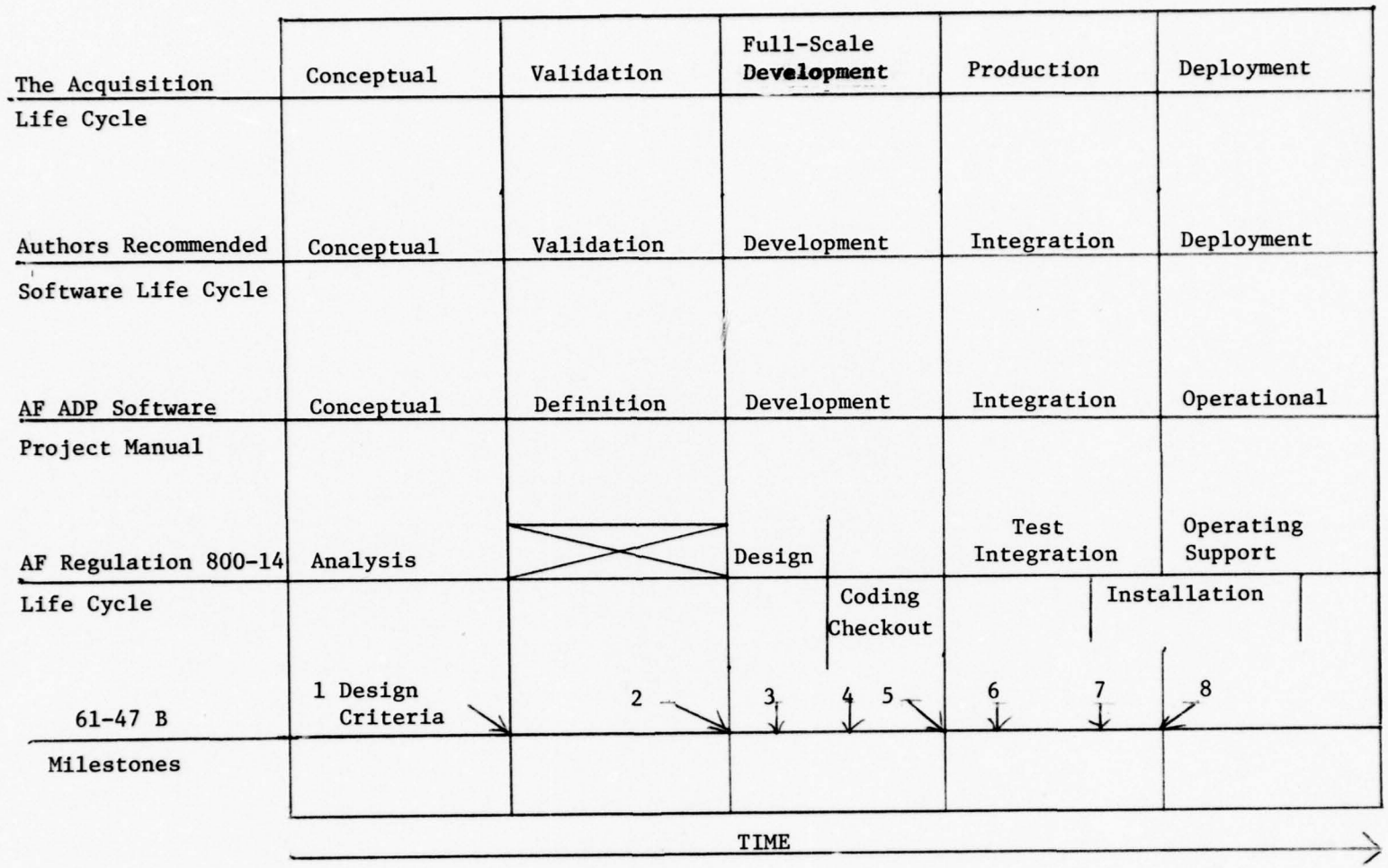



FIGURE 1

LIFE CYCLES

A life cycle enables the manager to determine what management steps need to be taken and when to take these steps.

A specific policy from DOD Directive 5000.29 is as follows:

> Specific milestones to manage the life cycle development of computer system and support software will be used to ensure the proper sequence of analysis, design implmentation, integration, test, documentation, operation, maintenance, and modification. These milestones will include specific criteria that measure their attainment (12:3).

## Life Cycle Phase Considerations

In order to be successful in acquiring software, there are many items to consider in each phase. Based on the author's experience a few items will be discussed below that appear to be significant.

In the conceptual phase the key item is to properly define the functional performance requirements for the computer programs. The needs and mission requirements of the user must be properly stated in a design criteria document or general specification. To properly accomplish this phase the user must be involved. This phase must be properly accomplished or the next phases will either fail or become excessively costly.

Also in this phase the maintenance concept should begin to be formulated. A maintenance concept is as important for software as it is for hardware. Who is going to modify the computer programs in the deployment phase? Who is going to analyze problems and program solutions to the problems? In-service personnel? Contractor personnel? How is the configuration of the operational programs going to be controlled? If in-service personnel are going to maintain the software then appropriate documentation

and training must be provided by the contractor. These items must be written into the contract for the development phase. Planning should begin in the conceptual phase and an initial maintenance concept should be formulated.

In the validation phase a key item to be considered is the hardware/software interface. What size computer is going to be used? How much core does the computer have? Can additional core be added, if needed? If not, does the proposed system present an undesireable risk? How fast is the computer? Is the computer fast enough to accomplish any required real time processing? Also decisions regarding the type of language must be made. Is a High Order Programming Language going to be used? Or, is a machine language going to be used? The risk areas must be identified, and a plan for their resolution must exist by the end of the validation phase.

The development phase seems to have two key items that lead to success. The first is to have well run Preliminary Design Reviews and Critical Design Reviews. If possible, the user should take part is these reviews. The second key item is to keep the number of changes to be a minimum. A development schedule cannot be maintained if changes are constantly made to the allocated baseline. At times it is absolutely necessary to freeze the baseline and hold all changes to a later model.

The integration phase consists of integration and testing of the computer programs in the operational environment. This testing is either accomplished by the developer or by an independent test organization

(inservice or independent contractor). Therefore, the first decision to be made is to decide who is going to accomplish the tests. The key to a successful integration phase is to make the tests as realistic and close to the operational situation as possible. For a large program all possible situations cannot be tested (cost prohibits this); thus good judgement must be made regarding what situations are tested. During the testing, configuration control must be maintained. It is absolutely necessary to know what program configuration is being tested.

More money can be spent in the deployment phase than any other phase if the computer programs are constantly changed. Computer programs can be easily changed; however, each change can cause unknown problems in other parts of the program. Therefore, each change needs to be tested. Program changes that correct known problems must also be tested as they could cause unknown problems. Each change made to improve the program should be fully justified before it is approved. All of the above can be summarized in two simple statements: Configuration control is vital for proper software management in the deployment phase. Testing of software changes must occur before they are added to the operational software.

## SECTION III

## CONFIGURATION MANAGEMENT

### General Comments

Most readers are aware how the discipline of configuration management is used in the management of hardware systems. Some readers are not aware that the same discipline can be and should be applied to the management of software systems. The principles of configuration management are used by commercial concerns as well as the military. When you order a part to repair your car, you receive a part that fits because the principles of configuration management are used by the automobile industry. When a commercial firm uses a computer through a time sharing arrangement, they get results from this rented time. To insure results, the configuration of the computer programs loaded on the computer are known and controlled. When the students at DSMC use the computer terminals for System X analysis, the output from the computer is predictable by the instructor because the computer programs loaded on the computers are consistant with the operational guide provided to the students. These are examples of applying configuration management to software.

DCD Directive 5000.29 directs that the discipline of configuration management be applied during software acquisitions for major programs. A specific policy statement from the directive is as follows:

> Defense system computer resources, including both computer hardware and computer software will be specified and treated as configuration items (12:2).

According to the definition in Section I, Configuration management

consists of three functions: configuration identification, configuration change control, and configuration status accounting. Before further discussion of these functions a discussion of baseline management is appropriate. Certain questions need answering. What criteria do you identify against? What do you change from?

## Baseline Management

The definition of baseline management as found in AFR 800-14 is:

> A fundamental concept associated with engineering management is the use of a series of configuration management baselines which aid in assuring an orderly transitional from one major decision point to the next throughout the system acquisition life cycle (1: 4-1).

Most documents list three baselines: functional, allocated, and product. These baselines can be directly related to the life cycles as discussed in section II. The functional baseline marks the end of the conceptual phase and the start of the validation phase. The functional baseline is established by the system specification (Type A). The allocated baseline marks the end of the validation phase and the start of the development phase. The allocated baseline is established by a development specification (Type B5). The product baseline marks the end of the development phase and the start of the integration phase. The product baseline is established by the product specification (Type C5).

The Air Force ADP Software Project Manual recommends the use of a fourth baseline for software management. The author agrees that a fourth baseline is needed. This baseline can be called the operational baseline. The operational baseline marks the end of the integration phase and the

start of the deployment phase. The operational baseline is established by the product specification plus a list of all discrepancies against the software. A discrepancy describes a problem that prevents the computer program from totally meeting some part of the product specifications. The discrepancies are of the type that can be "worked around" during the deployment phase. If the discrepancy (ies) is (are) severe enought to prevent the operational use of the program, the computer program must be returned to the programmer and the problems must be corrected. This will also require a partial or total reaccomplishment of the testing that is accomplished in the integration phase.

It is almost impossible to deliver software without discrepancies. Most large computer programs will have several discrepancies listed against them at the end of the integration phase. Schedules usually require the advancement to the deployment phase before all discrepancies are resolved. This is the reason an operational baseline is needed. During the deployment phase a computer program will advance from one operational baseline to other operational baselines as software maintenance is accomplished. Each one of these baselines must be properly controlled using configuration management procedures.

## Configuration Identification

Configuration identification provides the specific technical description of an item at any point in time. This identification is accomplished by using baseline documentation. As the software development progresses through the life cycle, the description of the computer program

becomes increasing detailed.

Baseline documentation used for configuration identification includes: system/subsystem specifications, development specifications, product specifications, interface specifications, and computer data requirements documents. Interface specifications describe either software to software or software to hardware interfaces.

Data requirements documents list and define data elements that the computer program must handle. The user must be involved in specifying these data elements.

Specification trees are used to identify computer program. Computer programs can be broken down to lower levels. At the bottom level the name routine is used. Two or more routines can be combined to form a module. Two or more modules can be combined into a computer program component. Two or more computer program components can be combined into a computer program configuration item (CPCI). A numbering system is needed to control and to facilitate traceability of CPCIs through out a program's lifetime.

## Configuration Control

Configuration control is the systematic evaluation, coordination, approval or disapproval, and implementation of all approved changes in the configuration of a CPCI after formal establishment of the baseline. Both the computer program and the documentation that describes the computer program must be controlled.

Engineering changes to hardware are classified as Class I or Class II changes. These same classifications can be used in configuration control of software. Class I changes are those changes that change a techni-

cal description listed in a baseline specification. Class II changes are changes that are not Class I changes. Class II changes in software management are those made to non baseline documentation such as changes to the users manual. This author also recommends that any program change that is made to correct a discrepancy while in the deployment phase be listed and controlled as a Class II change. Computer Program changes that correct discrepancies must be properly integrated and tested before they are allowed to be incorporated into the master operational computer program. Configuration control in the deployment phase is absolutely necessary. I recommend that formal configuration control boards to control software be established at the same point in the acquisition cycle as the configuration control board is established to control hardware. This is normally at the time of critical design review.

Configuration Status Accounting

Configuration status accounting is the recording and reporting of a CPCI's initially approved configuration identification and of all proposed and approved changes to the configuration identification. Status accounting begins when the first specification for a computer program is approved and released and continues throughout the life cycle of the software system.

Configuration status accounting must also include recording and reporting of all software problems (discrepancies). Records must contain traceability of all software problems to their corrective action. If, in the deployment phase, more thn one software model is in use, then configuration status accounting and reporting of each model must occur.

SECTION IV

TECHNICAL REVIEWS AND AUDITS

## General Comments

During the management of a software project, several formal reviews and audits should be held with the contractor(s). These reviews begin in the conceptual phase and continue through the integration phase. These reviews and audits need to be directed by appropriate words in the contract. As the formal reviews and audits will not provide all the information needed to manage a software project, the formal reviews must be augmented by several additional meetings between the contractor and the program management organization.

In this section the following reviews will be discussed: System Requirements Review (SRR), System Design Review (SDR), Preliminary Design Review (PDR), and the Critical Design Review (CDR). Also the Functional Configuration Audit (FCA) and the Physical Configuration Audit (PCA) will be examined. Many persons have experience using these reviews and audits in hardware projects. These same reviews and audits can be easily adapted and used to manage software.

## Technical Reviews

Most of the following definitions, comments, and discussion comes directly from MIL-STD - 1521 (20:4).

The objective of the SRR is to ascertain the adequacy of the contractor's efforts in defining system requirements. It will be conducted when a significant portion of the system functional requirements have been

established. The SRR should be conducted in the validation phase if the contractor has not been involved in the conceptual phase. The user should be involved in this review if possible.

The objective of the SDR is to evaluate the optimization, traceability, correlation, completeness, and risk of the allocated baseline. The SDR should be the final review before submittal of the validation phase products. This review should be in sufficient detail to insure a technical understanding between the contractor and the program management organization. The SDR should result in the identification of all computer programs required throughout the system. The computer programs consist of operational programs, maintenance/diagnostic programs, test/debug programs, exercise and analysis programs, simulation programs, compliers and assemblers. The computer program language to be used should be identified in the SDR. Finally all computer facilities needed to support the development of computer programs in the next phase should be identified.

The objective of the PDR is to evaluate the progress and technical adequacy of the selected design approach, to determine the compatibility of the design approach with the performance requirements of the CPCI development specification, and to determine the compatibility between the CPCI and other hardware or software items. The PDR at approximately the $\frac{1}{4}$ point in time of the development phase.

According to Mil-STD-1521, the following should be considered at PDR.

Computer Program functional Flow. This information shall

be completed to the level of flow charting which identifies the allocation of computer program components to functions and depicts the sequence of operation within the system functional flow.

Storage Allocation Charts. This information shall be detailed for the CPCI as a whole, describing the manner in which available storage is allocated to individual computer programs. Timing, sequencing requirements, and relevant equipment constraints used in determining the allocation are to be included.

Control Functions Description. A description of the executive control and start/recovery features for the computer program system shall be available, including method of initiating system operation and features enabling recovery from system malfunction.

Structure and Organization of the Data Base. The data base description shall be completed to a level which identifies data types and characteristics, structure layout, and allocation of data storage (20:19).

## Formal Audits

Most of the following definitions, comments, and discussion come directly from MIL-STD-1524 (20:4).

The FCA is held to verify that each CPCI has achieved the performance specified in the development specification (allocated baseline). Then verification is accomplished by examining and reviewing test data. The FCA is a prerequisite to acceptance of the development effort.

The PCA is a formal examination of the coded version of a CPCI against its technical documentation, and of the configuration management records to establish the product baseline. The PCA should not be conducted unless the program management organization has the final draft of the product specification. The PCA includes a detailed audit of the product specification, including its flow charts, listings, and design narrative.

SECTION V

MANAGEMENT OF DATA

General Comments

A hardware system consists of hardware, documentation, personnel, etc. A software system consists of the computer software, documentation, personnel, etc. Data (technical and status) is required to manage the acquisition of software and to support that software in the deployment phase. A software system is no more supportable without documentation than a hardware system would be without a technical manual.

The computer program, just like a hardware item, is acquired using a contract which includes a statement of work (SOW) and a contract data requirements list (CDRL). The computer program is produced and delivered in accordance with the SOW. Documentation to support the computer program is delivered in accordance with the CDRL. The DOD has an authorized data list (ADL) which identifies standard data item descriptions (DIDS) for us in acquiring data from contractors. The DIDS should be tailored to actual requirements.

Why Document?

DOD Directive 5000.29 directs that support items needed to maintain computer programs be specified in the contract and delivered. A specific policy statement from the directive is as follows:

> Unique support items required to cost effectively develop and maintain the delivered computer resources over the systems life cycle will be specified as deliverable, with DOD acquiring rights to their design and/or use. Examples of such support items are compilers, environmental simulators, documenta-

tion aids, test case generators and analyzers, and train-aids(12:3).

Volume II of AFR 800-14 states that each phase of development should be documented. It lists the needs for documentation as follows:

> Documentation is needed during development in order to track progress and provide information for management visibility and decision making. For example, specifications are required to describe the computer equipment and computer programs in terms of design performance, configuration and test requirements.
>
> Documentation is needed to enable proper life cycle support and maintenance of the computer resources. This documentation should be maintained and updated, as required, for each change to the system or system operation (1:7-1).

Documentation is also important to a software system because of a unique software situation. The programming of software is somewhat an "art". No two programmers would program a software routine the same. Therefore, how the routine is programmed must be documented. A few key programmers could have accomplished a significant portion of the programming for any software system. Their efforts must be documented to prevent the possibility of a personnel turnover that would cause an unacceptable risk to the program.

## Software Data Items

Some of the same data items that are required from a hardware contractor will be required from a software contractor. Examples are: Contract Fund Status Reports, Configuration Management Plan, Engineering Change Proposals, Specification Change Notices, Contract Work Breakdown Structure. System Specification, Development Specifications, and Test Plans.

Some of the support documentation required to provide additional design, test and operation information on a software program are listed below. Generally all of these will be produced while developing a software system.

Data Requirements Document. Lists and defines computer data elements that the computer program must handle, including data elements that the user must supply (3:4-15).

Data Base Specification. Specifies storage allocation and data base organization in sufficient detail to permit programmers to code the computer program and to generate required files, tables, dictionaries, and directories (3:4-16).

Computer Operators Manual. Specifies in detail the external chacteristics of a program and the procedure for initiating, executing, and terminating it. Describes each computer program function, including inputs, outputs, relationship with other functions, and the operating instructions for running the function, with limitations, restrictions, error messages, probable cause, recovery procedure, methods for estimating run time, etc. (3:4-17).

Program Maintenance Manual. Contains detailed instructions and procedures for maintaining and updating the software product during the operational phase (3:4-17).

Programmer Notebooks. Provides a development record and traceability data relative to the design, debugging, checkout, and integration of routines, modules, and functions of the CPCI. (Note: This item might not be a delivered item) (3:4-18).

SECTION VI

SUMMARY

## Conclusions

The Department of Defense and the Military Services are very concerned with the acquisition of software. This is evident by the number of studies authorized. Studies have been conducted by the Joint Logistics Commanders Software Reliability Work Group, United States Air Force Project Rand, Industrial College of the Armed Forces, United States Air Force Project Pacer Flash, and the John Hopkins University Applied Physics Laboratory.

Many directives, regulations and standards are currently published that guide the acquisition of software. Guidance provided in these documents is generally good and consistent. Some of the documents are hard to read and understand, especially for someone without prior knowledge of software management.

Many disciplines that have been applied to hardware acquisition can be adapted and applied to software acquisition. However, they generally need to be tailored to fit the exact situation (embedded or stand alone software project, small or large size procurement).

The life cycle can be used to manage software. The names used for each phase can vary with the situation. A software maintenance concept should be identified early in the life cycle. Software maintenance will be required in the deployment phase.

Configuration management should be used as an aid in the acquisition of computer programs. Configuration management must continue to be

applied in the deployment phase.

Technical reviews and audits provide the program manager with the needed information required to make management decisions. The same general reviews and audits that are used in hardware acquisition should be used in software acquisition.

Procument of data to support a computer program is necessary; without this data (documentation) the systems would be incomplete. Documentation is needed to manage during the acquisition phases and to support the system during the deployment phase.

Software acquisition can be properly managed. The necessary disciplines are available for use.

## Recommendations

1. That Program Managers use the same disciplines to manage software as they use to manage hardware.
2. That directives, regulations and standards written to cover both software and hardware be carefully analyzed to assure that software is adequately and correctly discussed.
3. That DSMC discuss the management of software acquisition more. Many of the engineering disciplines discussed in the Systems Engineering/Logistics Management Course apply to software as well as hardware. Yet, software is rarely used as an example. The two hour block in Fundamentals of Program Management should be expanded to four hours.

BIBLIOGRAPHY


1. Acquisition and Support Procedures for Computer Resources in Systems, AFR 800-14, Volume II, Department of the Air Force, Washington, DC., 26 September 1975.

2. A Guide for Program Management, AFSCP 800-3, Department of the Air Force, Andrews Air Force Base, DC., 9 April 1976.

3. Air Force ADP Software Project Manual, Draft Manual, Department of the Air Force, Washington, DC., June 1975.

4. Change Control Procedures for Automatic Data Processing Systems, Report No. TOR-269 (4110-01) - 38, Reissue C., The Aerospace Corporation, El Segundo, Ca., 1 February 1972.

5. Computer Program Subsystem Development Milestones, SSD Exhibit 61-47B, 1 April 1966.

6. Configuration Management of Software Programs, School of Engineering and Applied Science, George Washington University, Washington, DC., June, 1975.

7. Dreznes, S.M., Shulman, Hl, Ware, W.H., Smith, G.K., Davis, M.R., Reinstedt, R.N., Turn, R., The Computer Resources Management Study, Executive Summary, R-1855-PR, Rand, Santa Monica, Ca., September 1975.

8. Findings and Recommendations of the Joint Logistics Commanders Soft ware Reliability Work Group, Volume I, Executive Summary, Headquarters Air Force Systems Command, Andrews AFB, DC., November 1975.

9. Findings and Recommendations of the Joint Logistics Commanders Software Reliability Work Group, Volume II, Supporting Technical Information, Headquarters Air Force Systems Command, Andrews AFB, DC., November 1976.

10. Gansler, J.S., Deputy Assistant Secretary of Defense (OASD/I8L), "Comment", Defense Management Journal, Vol. II, No. 4, October 1975.

11. Kossinakoff, A., Sleight, T.P., Prettyman, E.C., Park, J.M., Hazon, P.L., DOD Weapon Systems Software Management Study, The Johns Hopkins University Applied Physics Laboratory, Laurel, Md., June 1975.

12. Management of Computer Resources in Major Defense Systems, DOD 5000.29, Department of Defense, Washington, DC., 26 April 1976.

13. Management of Computer Resources in Systems, AFR 800-14, Volume I, Department of the Air Force, Washington, DC., 12 September 1975.

14. Military Standard Configuration Control-Engineering Changes, Deviations and Waivers, MIL-STD-480, Department of Defense, Washington, DC., 30 October 1968.

15. Military Standard Configuration Control-Engineering Changes, Deviations and Waivers (Short Form), MIL-STD-481, Department of Defense, Washington, DC., 30 October 1968.

16. Military Standard Configuration Management Practices for Systems, Equipment, Munitions and Computer Programs, MIL-STD-483, Department of the Air Force, Washington, DC., 31 December 1970.

17. Program Management, AFR 800-2, Department of the Air Force, Washington DC., 16 March 1972.

18. Project Pacer Flask, Vol I, Executive Summary and Final Report, Air Force Logistics Command, WPAFB, Ohio, 28 September 1973.

19. Software Management Conference, Abridged Proceedings, First Series, Los Angeles, Section AIAA, Anaheim, Cal, 1976.

20. Technical Reviews and Audits for Systems, Equipment and Computer Programs, MIL-STD-1521, Department of Defense, Washington, DC., 1 September 1972.

21. Zempolich, B.A., An Analysis of Computer Software Management Requirements for Operationally Deployable Systems, Volume II, Industrial College of the Armed Forces Research Fellowa Program, Department of the Navy, April 1975.